```sql
/* SubQuery 개념
하나의 쿼리가 다른 쿼리에 포함되는 구조
다른 쿼리에 포함된 내부쿼리(서브쿼리)는 외부쿼리(메인 쿼리) 사용될 값을 반환하는 역할

- WHERE, HAVING 절에서 조건으로 => 단일행/다중행 서브쿼리
- FROM 절에서 테이블처럼  => 인라인 뷰 (Inline View)
- SELECT 절에서 컬럼처럼  => 스칼라 서브쿼리

<종류/사용 연산자>
- 단일행서브쿼리(단일열, 다중열): =, <, >
- 다중행서브쿼리(단일열, 다중열): IN, ANY, ALL

단일행과 다중행에서 사용할 수 있는 연산자가 다르다.
**/


-- 나승원 사원의 부서번호를 조회한다면?

SELECTEMP_NAME, DEPT_ID
FROMEMPLOYEE
WHERE EMP_NAME = '나승원';

-- 나승원이 속한 부서직원의 정보를 조회하라.

SELECT*
FROMEMPLOYEE
WHERE DEPT_ID = ( SELECTEMP_NAME, DEPT_ID
    FROMEMPLOYEE
    WHERE EMP_NAME = '나승원');


/*3. "국어국문학과" 에 다니는 여학생 중 현재 휴학중인 여학생을 찾아달라는 요청이
들어왔다. 누구인가? (국문학과의 '학과코드'는 학과 테이블(TB_DEPARTMENT)을 조회해서
찾아 내도록 하자)*/


SELECT  DEPARTMENT_NO
FROM  TB_DEPARTMENT
WHERE DEPARTMENT_NAME = '국어국문학과';

SELECT  STUDENT_SSN
FROM  TB_STUDENT;

SELECTSTUDENT_NAME
FROM  TB_STUDENT
WHERE DEPARTMENT_NO = ( SELECT  DEPARTMENT_NO
     FROM  TB_DEPARTMENT
     WHERE DEPARTMENT_NAME = '국어국문학과') AND ABSENCE_YN = 'Y'

```

```sql
-- 나승원과 같은 직업이고 나승원보다 급여가 많은 사원의 정보를 조회하라.

SELECT*
FROMEMPLOYEE
WHERE EMP_NAME = '나승원';

SELECT*
FROMEMPLOYEE
WHERE JOB_ID = (SELECTJOB_ID,
    FROMEMPLOYEE
    WHERE EMP_NAME = '나승원')
 AND
 SALARY > (SELECTSALARY
    FROMEMPLOYEE
    WHERE EMP_NAME = '나승원');

-- 최소급여를 받는 사원 정보를 조회하라...


SELECTMIN(SALARY)
FROMEMPLOYEE;

SELECT*
FROMEMPLOYEE
WHERE SALARY = (SELECTMIN(SALARY)
    FROMEMPLOYEE);

-- 부서별 급여 총합

SELECT DEPT_ID, SUM(SALARY)
FROM EMPLOYEE
GROUP BY  DEPT_ID;


-- 부서별 급여 총합이 가장 높은 부서의 이름과 급여총합을 조회하라.

SELECTMAX(SUM(SALARY))
FROMEMPLOYEE
GROUP BY DEPT_ID;


SELECTDEPT_NAME, SUM(SALARY)
FROM  EMPLOYEE
JOINDEPARTMENT USING(DEPT_ID)
GROUP BY DEPT_ID, DEPT_NAME
HAVINGSUM(SALARY) = ( SELECTMAX(SUM(SALARY))
    FROMEMPLOYEE
    GROUP BY DEPT_ID);
```

```sql
98
99  -- 부서별 최소급여를 받는 사원을 조회하라...
100
101 -- 부서별 최소급여 (SUBQUERY) > 다중행함수
102 SELECT DEPT_ID, MIN(SALARY)
103 FROM EMPLOYEE
104 GROUP BYDEPT_ID;
105
106 -- 다중행 서브쿼리에서 사용할 수 있는 연산자 (IN, ANY, ALL)
107 -- 다중서브쿼리는 범위로 비교
108 -- IN :EQUALS 비교
109 -- < ANY , > ANY, < ALL, > ALL
110
111 -- 부서별 최소급여를 받는 사원을 조회하라.
112 -- 사원 정보 (NVL: 널값을 다른 값으로 대체) Q
113 SELECT *
114 FROM  EMPLOYEE
115 WHERE (NVL(DEPT_ID, ' '), SALARY) IN (SELECTNVL(DEPT_ID, '
116         MIN(SALARY)
117        FROMEMPLOYEE
118        GROUP BY DEPT_ID);
119
120
121
122 /*과장보다 많은 급여를 받는 대리의 급여를 조회*/
123
124 /*대리 급여*/
125 SELECTSALARY
126 FROMEMPLOYEE E
127 JOIN  JOB J ON(E.JOB_ID = J.JOB_ID)
128 WHERE JOB_TITLE = '대리';
129
130 /*과장 급여*/
131 SELECTSALARY
132 FROMEMPLOYEE E
133 JOIN  JOB J ON(E.JOB_ID = J.JOB_ID)
134 WHERE JOB_TITLE = '과장';
135
136
137 SELECTJOB_TITLE, SALARY
138 FROMEMPLOYEE E
139 JOINJOB J ON (E.JOB_ID = J.JOB_ID)
140 WHERE JOB_TITLE = '대리'
141 AND SALARY > ALL (SELECTSALARY
142     FROMEMPLOYEE E
143     JOIN  JOB J ON(E.JOB_ID = J.JOB_ID)
144     WHERE JOB_TITLE = '과장');
145
```

```sql

-- 상관관계 SUBQUERY (Correlated Subquery)
-- MAIN 실행 -> SUBQUERY 실행 -> MAIN 실행

-- EXISTS: 서브쿼리가 결과를 하나라도 반환하면 TRUE
-- NOT EXISTS: 서브쿼리가 결과를 하나도 반환하지 않으면 TRUE

-- 관리자를 구하라.

SELECT EMP_ID, EMP_NAME, '관리자' AS "구분"
FROM EMPLOYEE E
WHERE EXISTS ( SELECT NULL
   FROM EMPLOYEE
   WHERE E.EMP_ID = MGR_ID)
UNION
SELECT EMP_ID, EMP_NAME, '직원' AS "구분"
FROM EMPLOYEE E
WHERE NOT EXISTS( SELECT NULL
    FROM EMPLOYEE
    WHERE E.EMP_ID = MGR_ID)
ORDER BY 3;


-- SET OPERATOR: SET 연산자
-- 두 개 이상의 쿼리 결과를 하나로 결합시키는 연산자. 이질적인 데이터를 하나의 데이터로
-- 주의사항) 컬럼의 숫자가 일치, 각 컬럼의 타입이 일치, 컬럼의 숫자가 일치하지 않을 때

-- <종류>
-- UNION: 합집합 (중복제거)
-- UNION ALL: 합집합 (중복허용)
-- INTERSECT: 교집합
-- MINUS: 차집합 (Oracle에서 사용)
-- EXCEPT: 차집합 (PostgreSQL 등에서 사용)

SELECT EMP_ID,
  ROLE_NAME
FROM EMPLOYEE_ROLE
MINUS
SELECT EMP_ID,
  ROLE_NAME
FROM  ROLE_HISTORY;

-- INLINE VIEW: FROM절에서 명시되는 서브쿼리. VIRTUAL TABLE

-- 직급별 평균급여를 조회하라

SELECT JOB_ID, TRUNC(AVG(SALARY),-5) AS JOBAVG   /*TRUNC(절삭):
FROM  EMPLOYEE
GROUP BY JOB_ID;
```

```sql

--Q
-- 직급별 평균급여를 받는 사원의 사원정보를 조회하고 싶다면?

SELECT EMP_NAME,
  JOB_TITLE,
  SALARY
FROM ( SELECT JOB_ID, TRUNC(AVG(SALARY),-5) AS
   FROM  EMPLOYEE
   GROUP BY JOB_ID ) V
JOIN EMPLOYEE E ON(V.JOBAVG = E.SALARY
    AND
    NVL(V.JOB_ID,' ') = NVL(E.JOB_ID, ' '))
LEFT JOIN JOB J ON(E.JOB_ID = J.JOB_ID);



/*14. 춘 기술대학교 서반아어학과 학생들의 지도교수를 게시하고자 한다. 학생이름과
지도교수 이름을 찾고 만일 지도 교수가 없는 학생일 경우 "지도교수 미지정'으로
표시하도록 하는 SQL 문을 작성하시오. 단, 출력헤더는 ,학생이름', ,지도교수'로
표시하며 고학번 학생이 먼저 표시되도록 한다. */
/*다시 풀기*/


SELECT*
FROM TB_STUDENT
WHERE COACH_PROFESSOR_NO IS NULL;

SELECT DEPARTMENT_NO
FROM TB_DEPARTMENT
WHERE DEPARTMENT_NAME = '서반아어학과';

SELECT STUDENT_NAME AS "학생이름",
 NVL(PROFESSOR_NAME,'지도교수 미지정') AS "지도교수"
FROM TB_STUDENT S
LEFT JOIN TB_PROFESSOR P ON (S.COACH_PROFESSOR_NO =
WHERE S.DEPARTMENT_NO = ( SELECT DEPARTMENT_NO
     FROM TB_DEPARTMENT
     WHERE DEPARTMENT_NAME = '서반아어학과')
ORDER BY ENTRANCE_DATE;



/*15. 휴학생이 아닌 학생 중 평점이 4.0 이상인 학생을 찾아 그 학생의 학번, 이름, 학과
이름, 평점을 출력하는 SQL 문을 작성하시오.*/

/*강사님 풀이*/
```

```sql

SELECT STUDENT_NO AS "학번",
 STUDENT_NAME AS "이름",
 DEPARTMENT_NAME AS "학과 이름",
 AVG(POINT)
FROM TB_STUDENT S
JOIN TB_DEPARTMENT D USING(DEPARTMENT_NO)
JOIN TB_GRADE G USING (STUDENT_NO)
WHERE ABSENCE_YN = 'N'
GROUP BY STUDENT_NO, STUDENT_NAME, DEPARTMENT_NAME
HAVING AVG(POINT) >= 4.0
ORDER BY STUDENT_NO;


/*16. 환경조경학과 전공과목들의 과목 별 평점을 파악할 수 있는 SQL 문을 작성하시오.*/
/*다시 풀기*/

SELECT DEPARTMENT_NO
FROM TB_DEPARTMENT
WHERE DEPARTMENT_NAME = '환경조경학과';
/*환경조경학과 DEPARTMENT_NO = 034*/

SELECT *
FROM TB_CLASS
JOIN TB_DEPARTMENT USING (DEPARTMENT_NO)
WHERE DEPARTMENT_NO = '034' OR CLASS_TYPE LIKE '전공%';


SELECT C.CLASS_NO, C.CLASS_NAME, AVG(POINT)
FROM TB_DEPARTMENT D
JOIN TB_CLASS C ON (D.DEPARTMENT_NO = C.DEPARTMENT_NO)
JOIN TB_GRADE G ON (C.CLASS_NO = G.CLASS_NO)
GROUP BY D.DEPARTMENT_NAME, C.CLASS_NO, C.CLASS_NAME,
HAVING D.DEPARTMENT_NAME = '환경조경학과' AND CLASS_TYPE LIKE
ORDER BY CLASS_NO;

/*17. 춘 기술대학교에 다니고 있는 최경희 학생과 같은 과 학생들의 이름과 주소를 출력하는
SQL 문을 작성하시오*/

SELECT DEPARTMENT_NO
FROM TB_STUDENT
WHERE STUDENT_NAME = '최경희'

SELECT STUDENT_NAME, STUDENT_ADDRESS
FROM TB_STUDENT
WHERE DEPARTMENT_NO = ( SELECT DEPARTMENT_NO
    FROM TB_STUDENT
```

```sql
    WHERE STUDENT_NAME = '최경희');

/*18. 국어국문학과에서 총 평점이 가장 높은 학생의 이름과 학번을 표시하는 SQL 문을
작성하시오.*/

SELECT DEPARTMENT_NO
FROM TB_DEPARTMENT
WHERE DEPARTMENT_NAME = '국어국문학과';
/*국어국문학과 DEPARTMENT_NO = 001 */

SELECT S.DEPARTMENT_NO , AVG(POINT)
FROM  TB_GRADE G
JOIN TB_STUDENT S ON (S.STUDENT_NO = G.STUDENT_NO)
GROUP BY S.STUDENT_NO, S.DEPARTMENT_NO
HAVING S.DEPARTMENT_NO = ( SELECT DEPARTMENT_NO
    FROM TB_DEPARTMENT
    WHERE DEPARTMENT_NAME = '국어국문학과');

SELECT *
FROM TB_CLASS
WHERE DEPARTMENT_NO ='001';

SELECT STUDENT_NAME, STUDENT_SSN
FROM TB_STUDENT


/*19. 춘 기술대학교의 "환경조경학과"가 속한 같은 계열 학과들의 학과 별 전공과목 평점을
파악하기 위한 적절한 SQL 문을 찾아내시오. 단, 출력헤더는 "계열 학과명",
"전공평점"으로 표시되도록 하고, 평점은 소수점 한 자리까지만 반올림하여 표시되도록
한다.*/


SELECT CATEGORY
FROM TB_DEPARTMENT
WHERE DEPARTMENT_NAME = '환경조경학과';
/*자연과학 계열*/



SELECT DEPARTMENT_NAME AS "계열 학과명",
  ROUND(AVG(POINT),1) AS "전공평점"
FROM  TB_DEPARTMENT D
JOIN TB_CLASS C USING (DEPARTMENT_NO)
JOIN TB_GRADE G USING (CLASS_NO)
WHERE CATEGORY = (SELECT CATEGORY
    FROM TB_DEPARTMENT
    WHERE DEPARTMENT_NAME = '환경조경학과')
GROUP BY DEPARTMENT_NAME
```

```
339 ORDER BY 1;
340
341
342
343
344
345
346
347
348
349
350
351
352
```